AFIT/GIR/ENG/95D-4

A NEURAL NETWORK APPROACH TO THE PREDICTION AND CONFIDENCE ASSIGNATION OF NONLINEAR TIME SERIES CLASSIFICATIONS

THESIS

Erin S. Heim
Captain, USAF

AFIT/GIR/ENG/95D-4

Approved for public release; distribution unlimited

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government.

AFIT/GIR/ENG/95D-4

A NEURAL NETWORK APPROACH TO THE PREDICTION AND CONFIDENCE ASSIGNATION OF NONLINEAR TIME SERIES CLASSIFICATIONS

THESIS

Presented to the Faculty of the School of Logistics and Acquisition Management

Air Education and Training Command

In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Information Resource Management

Erin S. Heim, B.S.

Captain, USAF

December 1995

Acknowledgments

I would first like to thank my thesis advisor, Dr. Steven Rogers, for his infinite patience and technical assistance in support of my research. In addition, I'd like to thank my readers, Dr. Kim Campbell and Dr. Dennis Quinn, who helped to ease the burden of the thesis process by providing expert assistance in the writing of this thesis.

I would also like to thank my sponsor, who more importantly is my Dad. Thank you for putting up with all my questions about the stock market data and calculations. I truly had no idea of the breadth of information available. I'm truly impressed and proud of your technical expertise and insight. This research began with your suggestion and would be nowhere without your market insight. Despite some set backs (intellectual and monetary), you were always there to encourage me.

My daughter, Alexandria, deserves a great deal of thanks. I know it was not easy for you being without your Mommy when she "had to study tonight".

Finally, I would like to thank my husband, Rich. His never-ending support of my career and education has given me the opportunity to excel beyond my wildest dreams.

Erin S. Heim

Table of Contents

Page

List of Figures

List of Tables

Abstract

This thesis uses multiple layer perceptrons (MLP) neural networks and Kohonen clustering networks to predict and assign confidence to nonlinear time series classifications. The nonlinear time series used for analysis is the Standard and Poor's 100 (S&P 100) index. The target prediction is classification of the daily index change. Financial indicators were evaluated to determine the most useful combination of features for input into the networks. After evaluation it was determined that net changes in the index over time and three short-term indicators result in better accuracy. A back-propagation trained MLP neural network was then trained with these features to get a daily classification prediction of up or down. Next, a Kohonen clustering network was trained to develop 30 different clusters. The predictions from the MLP network were labeled as correct or incorrect within each classification and counted in each category to determine a confidence for a given cluster. Test data was then run through both networks and predictions were assigned a confidence based on which cluster they belonged to. The results of these tests show that this method can improve the

accuracy of predictions from 51% to 73%. Within a cluster accuracy is near 100% for some classifications.

A NEURAL NETWORK APPROACH TO THE PREDICTION AND CONFIDENCE ASSIGNATION OF NONLINEAR TIME SERIES CLASSIFICATIONS

I. Introduction

Background

Predicting the future based on what has happened in the past is a problem that has fascinated mathematicians and astrologers alike. With a linear time series the problem is quite trivial; the equation can generally be determined so that it will yield a satisfactory answer in most cases. Unfortunately, most of the problems of interest and usefulness are not linear; they are nonlinear. Nonlinear time series currently under study in the Air Force include aircraft position prediction, missile target prediction, and pilot head motion studies (Longinow, 1994).

It has been shown that statistical techniques, such as linear regression, are not very successful in making nonlinear predictions (Refenes, 1994). In addition, graphical and technical analysis are only marginally successful if the amount of data is not too overwhelming (Barr and Mani, 1994). The method that will be explored in

this thesis will incorporate the use of the rather youthful technology of artificial neural networks (Rogers & Kabrisky, 1993). In addition, we use a Kohonen Self-Organizing Map network which clusters the patterns into categories based upon the proximity of the features to each other.

Past samples of the time series are generally used to predict future time series values. These past samples are usually not the only data that is useful in prediction. For example, in predicting the change in the Dow Jones Industrial Average index, not only past values of the index are useful, but volume numbers and other index values such as the S&P 500 would also be helpful in predicting future closes of the Dow Jones index. Part of the problem is determining which information will aid prediction.

A common practice in training a neural network is to throw everything at it but the kitchen sink in the hopes that it will use only what is important. In contrast, using the technical expertise of analysts of the problem, extraction of useful features can be done more accurately. Similar to the knowledge acquisition process used to extract rules for an expert system (Mockler and Dologite, 1992), relevant features needed to train a neural network can be extracted from experts as well.

Problem Statement

This study will investigate the use of both MLP neural networks and Kohonen clustering networks to improve classification of daily changes in a nonlinear time series and provide a measure of confidence in that predicted classification.

Scope

A backpropagation trained MLP neural network was trained to classify daily changes in the closing index of the Standard and Poor's 100 (S&P 100). Data is available for many days from many years on the S&P 100, unlike some of the other nonlinear time series discussed earlier and was chosen for that reason. Once a network was successfully trained on the indicator data, a Kohonen clustering network was trained that clustered the data based on the similarity of the indicator values. Each cluster was then analyzed based on the accuracy of the predictions taken from the MLP network. A percentage was assigned to each cluster based on the cluster's accuracy in classifying up versus down. Test data was next run through the MLP and Kohonen networks to give a classification of up or down and a cluster number. Based on the prediction and the cluster that vector was assigned, a confidence percentage was determined. Results were

interpreted based on win/loss of dollars had this prediction instigated the purchase of a stock option.

Thesis Organization

The following chapter will discuss how neural networks are trained to make predictions. In addition, it will describe what a Kohonen Self-Organizing Map network and how it works. It will also discuss the various methods used to predict stock market behavior prior to this work. Chapter III will describe in detail the methodology that was developed in this research to improve the confidence in neural network predictions of nonlinear time series. The results of using this methodology in predicting S&P 100 index changes will be presented and discussed in Chapter IV. Chapter V summarizes the results and contributions of this research and suggests areas for future research.

Summary

A commonly used representation of a nonlinear time series is the S&P 100 index. This thesis will investigate whether the use of Kohonen Self-Organizing Map networks can improve the confidence in neural network predictions of nonlinear time series. The results of this investigation should be applicable to providing confidence in other neural network predictions.

II. Research Background

Introduction

Chapter I introduced the nonlinear time series problem and discussed the methodology that will be used in this research. This chapter will more clearly define what a neural network is and how it is typically applied to a nonlinear time series problem. In addition, a discussion of Kohonen Self-Organizing Maps will provide an understanding of its use in nonlinear time series prediction. Finally, because the data set used in this thesis is the S&P 100, this chapter will review the present neural network techniques to predict stock market prices for investment purposes.

Neural Networks

A relatively new technique that is used to predict nonlinear time series is the neural network. Neural networks are a branch of computer artificial intelligence. These networks are a simplistic computer simulation of the human brain. A computer neural network resembles the human brain in two aspects: 1. Knowledge is acquired by the network through a learning process; and 2. Interneuron connection strengths known as synaptic weights are used to store the knowledge (Haykin, 1994:52). The beauty of a

neural network is that it is continually learning as it processes. It 'fixes' its process based on the data or 'knowledge' it acquires.

The computer neural network is composed of processing elements or perceptrons which act similarly to the brain's neurons. To understand the processing of a computer perceptron, it is helpful to understand the biological neuron. A neuron is a nerve cell with all of its processes. There are many different types and classes of neurons in the human body; we will only discuss a generalized version of the neuron (Muller and Reinhardt, 1990:26-28). There are three parts to a nerve cell: cell body, dendrites and axon. The cell body contains the nucleus of the cell. Leading to the nucleus are dendrites. The dendrites conduct the impulses toward the nucleus. The axon conducts impulses away from the cell body. Many neurons or nerve fibers form nerve structures or networks. The connection between two neurons is the synapse. At this point, the axon of one neuron connects to the dendrite of another neuron and impulses can be passed along the neural pathway. If the nerve is stimulated at or above its threshold, then the neuron will fire. If it is not stimulated to its threshold level, then it doesn't fire. Thus, only one of two things

can occur, the neuron either fires or it does not (Azoff, 1994:14).

Signals come into synapses and are weighted and summed (Rumelhart, 1994:87). If the sum is greater than or equal to the threshold for the neuron, the neuron fires. These signals can be adjusted by activity in the nervous system. Threshold functions integrate the signals into an output (Rumelhart, 1994:87). Since the neuron either fires or does not, the rate of firing is more important than the amplitude of firing. Responses to inputs and how the system organizes itself in response to the error in output is what learning is all about (Azoff, 1994:14). The importance of this process is clear when you break the process down and compare it to a computer model.

When modeling synaptic activity in a computer, we start with the perceptron, sometimes called a processing element (PE). Various inputs or signals are given to a PE. They all go into the PE simultaneously, and the response is that it either fires or not. Each of these inputs is weighted with either an exciter or inhibitor based on its effect on the desired output. Excitement increases the probability that the PE will fire while the inhibitor has the opposite effect. The input is multiplied by its weight and then all of these products are summed to produce an output. The

output will either be below or meet and exceed the threshold of the PE, in which case, a signal is then generated. This signal is passed on to an activation function. The purpose of this function is to vary the signal before it goes to the transfer function. Next the transfer function takes the signal and processes it for a final output signal. Typically a sigmoid function (S-curve) is used rather than a linear or step function (Nelson and Illingworth, 1990). Transfer functions have a squashing role in restricting the possible neuron output, which takes a value that may lie in the range of $(-\infty, \infty)$ and constrains it to, typically, [0, 1] or [-1, 1] (Azoff, 1994:14).

It is important to note that unlike the nervous system neuron which either fires or does not (binary output), the perceptron's output is an analog number that is based on the number of pulses fired and the weight of the original input.

To simulate "learning", we can attach some memory to the PE and store results of previous trials, and then modify the weights as we go along to change the signals. This is where back propagation comes into play (Werbos, 1974). If you take the network's output signal and compare it to the actual output, you create feedback for the network and give the network a means to correct itself if it is wrong.

Most networks, as the word network implies, use many perceptrons. The first layer of perceptrons is the input layer; each perceptron receives direct input from each characteristic or feature chosen from the problem. The second or hidden layer receives the outputs from the first layer and subsequently feeds its outputs to the third or output layer. Figure 1 shows the architecture of a typical MLP backpropagation neural network. A network is considered fully connected when every output from one layer is passed along to every node in the next layer. If all outputs proceed to the next layer and network outputs are not compared to actual outputs, the network is feed forward. Should any outputs go to the preceding layers, it is a feedback network.

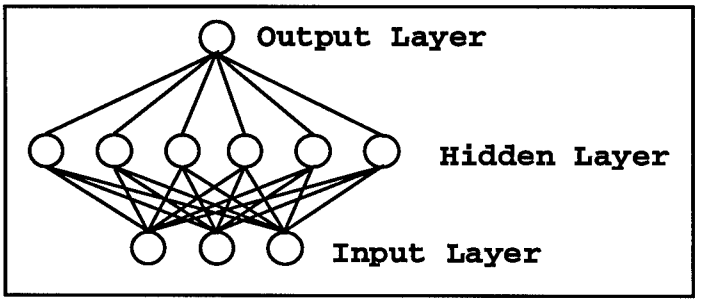

Figure 1. Typical Neural Network Architecture

The type of MLP backpropagation network used in this study is a classification neural network. A classification neural network has a neuron for each 'feature' or input into

the network for the input layer. The hidden layer is made up of how ever many neurons is deemed necessary to sufficiently classify the problem at hand. There is one output neuron in the output layer for each class in the problem. For example, the classes used in this study are Class 1 (up), for positive changes in the S&P 100 index and Class 2 (down), for negative changes in the index. The sigmoid transfer function plays a critical role in determining the output. Should the neurons weighted sum from the hidden layer land to the left of the center point of the curve, the output is classified in class 1 (up). Should it lie to the right of the center inflection point, the output is classified in class 2 (down).

Typically data is parsed into three subsets. The first subse is the training set which is used continuously by the network to adjust it weights and to learn the patterns in the data.

The second subset is the test set. The test set is periodically presented to the network, and an error rate is calculated based on the current state of the network at that point in training. This error rate will slowly get better and better as the network "learns" the training data and the training error decreases. But, at a point of network optimization, the test set error will begin to rise while

the training set error continues to fall. At this point the network is now adjusting itself to the specific training data in such a way that it is now "memorizing" the training set. By using the test set for a periodic check, the network can be stopped at the time when learning has reached a maximum.

The final subset is the production set. We hold the production set until the end of training to determine whether the network can generalize well against data it has never seen before. The production set gives you a true test of the generalizability and accuracy of your network if you plan to use it in the future on real-time data. If this error rate is acceptable, the network is ready to use against real-time data to make predictions.

Kohonen Networks and Unsupervised Learning

Unsupervised learning is a method of training a neural network without showing it correct outputs in sample training patterns. It relies on regularities or trends in the input signals and makes adaptations based on these.

Another name for this type of network is a "clustering" network. If you were able to plot in N-dimensional space the N features of the data set, you might see the data clustering together in certain areas. When training a clustering type network, it determines which patterns belong

in which cluster based on the proximity of the features in the patterns.

Commonly these classification networks are used when the number of classes are unknown. One problem is that we do not know the optimum number of classes the data can be clustered into. When training these networks we start with an initial number and based on the results add or subtract clusters.

The Kohonen Self Organizing Map network is the type of unsupervised learning network that is used in this study. "A mapper, in this context, is a mathematical transformation that takes input data vectors and maps them to output vectors" (Rogers & Kabrisky, 1991:61). The input data patterns are clustered based on their proximity in N-dimensional space where N is the number of inputs. The user tells the network the maximum number of clusters and the network will usually put the data into that number of categories (Ward Systems Group, 1993:216).

The learning process is somewhat different for Kohonen networks than for MLP backpropagation neural networks. Kohonen networks have only two layers, the input layer and the output layer; whereas the MLP backpropagation network has three layers (input, hidden, and output). The patterns are presented to the input layer, then propagated to the

output layer and evaluated. One output neuron is the "winner," i.e., the weight vector (all the weights) leading to this neuron is closer in N dimensional space to the input pattern than that of any other output neuron. The network weights are then adjusted during training by bringing this weight vector slightly closer to the input pattern. This process is repeated for all patterns for a user-specified number of epochs (Ward Systems Group, 1993:55).

Now that the discussion of neural and Kohonen networks is complete, an examination of the application of these technologies follows.

Selecting the Best Inputs For a Financial Neural Network

Currently investment managers have to assimilate great quantities of data in the marketplace. To do this general statistical measures which combine many indicators have been developed to minimize "information overload". But there are still many other quantitative and qualitative factors to consider in their analysis and final recommendation of where to invest.

A neural network can use various inputs to predict the outcome of a particular variable. The research done in this area has shown that it is better to predict market index change rather than absolute values. Also, it is important to select the proper inputs to form the basis of the network and to select the right time horizon (forecast a few minutes

into the future, or a few days, months, etc.). Through various experiments with input selection, Dean S. Barr and Ganesh Mani found that one should choose a few important indicators and then "telescope" these indicators out (Barr & Mani, 1994). In other words, select the indicator and then use data from various time periods (5, 10, 20 time periods back). Barr and Mani also found that the network gave the best results when it was only predicting about 5 to 10 days into the future, rather than a month or longer.

Barr and Mani trained a neural network using a data set of 182 trading days of which 164 were used in the training set and the remaining 18 were used as the production set to test the output (Barr & Mani, 1994). Once the inputs and time horizon were chosen, they conducted a sensitivity analysis on the inputs. They changed one input at a time and tested the percent change it had on the final output (called dithering). In this way, they could determine which indicator or input seemed to have the greatest effect on the predictive success of the net. Adjustments could then be made to further "prune" the inputs that should be included or excluded. This technique is used to gain confidence in the features selected for use in this thesis (Ruck, 1990).

Another study by Bernd Freisleben, suggests using other rather unique types of input data.

> While in other approaches the input data was exclusively based on stock prices, we also consider other important economical factors, namely a subset of

> those considered in the fundamental and technical analysis methods used by human analysts to make their investment decisions. (Freisleben, 1992)

Some of these factors included economic environment variables. Freisleben's network acts as a human analyst with the added ability to recognize patterns in the volumes of data that a human might not be able to assimilate.

In this thesis, Freisleben's approach to choosing data was employed. Through talking with a human analyst, the indicators that the analyst used to make predictions were simulated or directly added to the network as features.

High Transaction Costs in Neural Investing

LBS Capital Management, Incorporated in Clearwater, Florida currently invests over $600 million, half of which are pension assets which are chosen using neural network techniques (Elgin, 1994). LBS admits that some customers are uneasy about this type of investing, but results have been good.

> LBS has reported no loss year in stocks or bonds since the strategy was launched in 1986. Its mid-cap fund returns have ranged from 14.53% in 1993 to 95.60% in 1991, compared to the S&P 500, which returned 13.95% and 50.10%, respectively. (Elgin, 1994)

Thus, it appears that it is possible to use neural networks to effectively invest.

One characteristic, and perhaps downfall, to neural network investing is active trading. Turnovers in LBS ranged from 150 to 200%, and at another company using this

type of technology, turnover runs 300 to 400%. Mr. James Hall, engineer at Deere & Co. which has assigned $150 million of internally managed pension assets to an artificial intelligence investment program, says Deere pays 8 to 10% of its profits a year in transaction and commission costs (Elgin, 1994). In order to make money, the companies must have a high percentage rate of return. Based on this, investing based on a network's prediction requires a sufficient degree of confidence so that entrance and exit in and out of the market occurs only when necessary. Rather than constantly change investment strategy based on each network prediction (and consequently pay more transaction fees), it may be wiser to ascertain a desired degree of confidence and come up with longer term holding strategies, like those developed in this thesis rather than relying solely on the overall error rate of a neural network for confidence and making numerous trades.

Measurement of Success and Failure

It was noted with particular interest that Dan Murray used the dollar amount earned (or lost) by his model on the stock market in a given period of time to determine how success of his neural network (Murray, 1994). This measure is particularly useful with a classification network, where the usual means of determining success or failure is through percent accuracy. This percentage can be deceiving. For

instance, the network can be 60% successful, but if the days it incorrectly predicted caused larger losses based on the amount of index points incorrectly predicted, investments can still lose money using that network.

The results of this research will overcome this shortcoming. Not only will we show network overall accuracy, but also dollar wins/losses based on those predictions.

Conclusion

In reviewing the literature, there is an emerging interest in the use of neural network technology to invest "real" not "test" money profitably. All designers agree that the financial markets are extremely complex and change not only due to mathematical reasons, but also due to psychological reasons. These changes are neither structured nor linear. A neural network can deal with this lack of structure, but is only as good as the data input. Research in this area is unlimited based on the many possible inputs and architectures that can be used with a neural network.

In this research effort, various market indicators (e.g., net changes over time) are used to develop a backpropagation neural network to predict stock market changes. In addition, a Kohonen Self-Organizing Map is used to cluster the indicator data to offer a degree of confidence in the prediction made by the backpropagation

network. By offering a sufficient level of confidence in the prediction, entry and exit in and out of the market can be minimized. Chapter III will discuss the development of this methodology as well as its implementation. Chapter IV will use the dollar earned/lost technique to show relative success or failure of the methodology.

III. Methodology

Introduction

The previous chapter provided background on MLP neural networks and Kohonen networks, as well as a discussion regarding the use of these networks to predict a financial time series. This chapter will discuss the development of the method of using both the MLP and Kohonen networks to arrive at a prediction of a time series and determine percentage of confidence in the prediction.

Input Data

The data samples were provided by Decision Point, Inc. Values from three different indicators and S&P 100 closing prices were taken from July 1989 to the present. This comprised approximately 1580 trading days worth of indicator data that will be used to train and test the backpropagation network. The test sets were generated by selecting randomly 10% of the total patterns from July 1989 to present. Production sets were generated randomly as well, unless otherwise noted in the experiment. The Kohonen network clustered only the patterns in the training and test sets; the production set data was not used to derive confidence tables.

The target classification for the neural network is the delta between today's S&P 100 close and tomorrow's S&P 100

close. All indicator data are currently available from Decision Point, Incorporated (see Appendix A for references). Data was downloaded from the Decision Point Timing and Charts Forum on America Online.

The indicators were selected using a knowledge engineering approach (Mockler & Dologite, 1992:44) and are listed in Table 1. Discussions were held at length with Carl Swenlin, CEO of Decision Point, Inc., a stock market

Table 1

Network Features

List of Features
1. 1-Day Net Change in S&P 100 Index
2. 5-Day Net Change in S&P 100 Index
3. 21-Day Net Change in S&P 100 Index
4. McClellan Oscillator Index Value
5. McClellan Oscillator 1-Day Net Change
6. Yesterday's McClellan Oscillator 1-Day Net Change
7. Swenlin Oscillator Index Value
8. Swenlin Oscillator 1-Day Net Change
9. Yesterday's Swenlin Oscillator 1-Day Net Change
10. Short Term Volume Oscillator Value
11. STVO 1-Day Net Change
12. Yesterday's STVO 1-Day Net Change

technical analyst. Because it was determined that the target classification would be daily changes in the S&P 100 market index, the best short-term indicators were chosen. As a technical analyst of the market, the features that are analyzed to make this type of short-term daily prediction are the market trends (short, mid and long term) in conjunction with the values and trends of short-term market indicators.

It is important to note that when choosing which indicators to use, the MLP backpropagation neural network will only "see" one line of data at a time, it doesn't actually "remember" the data from the previous line. Thus, the inputs should be chosen so that all the information that the network will need to make an accurate prediction is in one line of data. The network will not automatically see trends in the index over time as it changes line by line.

To capture the market trend over the short to long term, three features were added to the neural network data input. First, the net change in the S&P 100 index from yesterday to today was added (1-Day Net Change). Second, the net change in the S&P 100 index over the last 5 days was added (5-Day Net Change). Finally, the net change in the S&P 100 index over the last month or 21 trading days was added (21-Day Net Change).

The three short-term indicators chosen were the McClellan Oscillator, Swenlin Oscillator, and Short Term Volume Oscillator. A discussion of each of these indicators follows. Each indicator had today's oscillator value added to the network as a feature. If t represents today and t-1 represents yesterday, etc.; then the change in each oscillator between t and t-1, as well as the change between t-1 and t-2 were also added as features.

McClellan Oscillator. The McClellan is a breadth-based indicator. This means it is derived from the daily advances minus declines on the New York Stock Exchange (Decision Point, 1995). This oscillator was invented in the 1960's by Sherman and Marion McClellan, and since then it has proven to be one of the most useful analysis tools in existence (Decision Point, 1995).

To calculate the McClellan Oscillator, calculate a 40 day exponential moving average (0.05 average) and a 20 day exponential moving average (0.1 average). After calculating the two averages each day, subtract the 40 day EMA from the 20 day EMA to get the McClellan Oscillator value. The following are the exact formulas:

Today's 40 Day EMA formula (1):

$$((TAD - P40)*0.05) + P40 = T40 \qquad (1)$$

where

TAD = Today's Advance Minus Decline

P40 = Prior Day's 40 Day EMA

T40 = Today's 40 Day EMA

Today's 20 Day EMA formula (2):

$$((TAD - P20)*0.10) + P20 = T20 \quad (2)$$

where

TAD = Today's Advance Minus Decline

P40 = Prior Day's 40 Day EMA

T20 = Today's 20 Day EMA

McClellan Oscillator formula (3):

$$T20 - T40 = \text{McClellan Oscillator} \quad (3)$$

Swenlin Trading Oscillator. The Swenlin Trading Oscillator (STO) is another breadth-based oscillator designed for use in short-term trading. The STO is a 5-day moving average of a 4-day exponential moving average (EMA) of the daily advances minus declines (A-D).

> The double smoothing of the short-term data results in a pretty reliable oscillator that usually tops near short term market tops and bottoms near short-term market bottoms. The stronger the market the less accurate the STO will be at picking tops, but STO bottoms in the area of -200 and below are fairly good predictors/confirmations of short-term market bottoms. (Decision Point, 1995)

To calculate the STO, first calculate the average value of advances minus declines for the last four days before beginning the exponential weighting. Then calculate the exponential average. The formula for the exponential moving average (EMA) is (4):

$$\underline{EMA} = (((\underline{A-D})-\underline{pdEMA})*0.5)+\underline{pdEMA} \qquad (4)$$

where

pdEMA = Prior Day's Average (Begin with simple moving average, thereafter pdEMA is an exponential average.)

A-D = Current day's advances minus declines.

All that remains is to calculate a 5-day simple moving average of the EMA to derive the Swenlin Trading Oscillator.

Short Term Volume Oscillator. The Short Term Volume Oscillator (STVO) summarizes climactic volume activity for a specific market index. The S&P 100 STVO is stable and represents what is happening in the broader market. The exact calculation method for the STVO is proprietary to Decision Point, Inc., but it can be said that it is derived from volume calculations for each stock making up the S&P 100 index (Decision Point, 1995).

The scale for the Dow Jones Industrial Average (DJIA) STVO is -30 to +30. As a result, the raw STVO for the S&P

100 is multiplied by 0.3 so that the result will fit the scale and can be compared to the DJIA STVO. The normal range for the S&P 100 STVO is +10 to -10.

Decision Point says that the STVO is most useful in picking bottoms because volume trends normally tend to spike at bottoms in concert with price. Because of this, extremely oversold STVO bottoms can give reliable, simultaneous confirmation of price bottoms.

> The STVO is not so useful at picking tops because STVO tops are not necessarily coincident with significant market tops; however, we can expect confirmation of tops by the STVO. (Decision Point, 1995)

Software

The neural network software that will be used to train, test and predict is NeuroShell 2. The software is developed and distributed by Ward Systems Group, Incorporated. The software requires the use of an IBM PC or compatible with an 80386 or higher processor, 4 megabytes of RAM, and about 5 megabytes of hard disk space.

The software supports both backpropagation MLP neural networks and Kohonen Self Organizing Map networks. It also supports other architectures that were not utilized in this study.

The backpropagation trained MLP neural network architecture used has 12 input nodes, a hidden layer of 44 perceptrons, and two output nodes (one for each class, up and down). There is one perceptron for each indicator in

Table 1 in the input layer. The hidden layer size was determined by the software based on the number of features and the amount of input data patterns. The Kohonen network also has 12 input nodes, no hidden layer, and 30 output neurons (one for each possible output category). The network adjusts the weights for the neurons in a neighborhood around the winning neuron until during the last training events the neighborhood is zero, meaning by then only the winning neuron's weights are changed (Ward Systems Group, 1993).

The scaling function for the first layer is linear. This means that the input data is normalized or 'squashed' into the interval [-1,1]. The activation function for both the hidden layer and the output layer is the standard logistic or sigmoid function. This function maps the outputs into the (0,1) range.

The MLP neural network will update its weights after each training epoch (one pass through the entire training set) known as a "batch" update. The network adds all of the weight changes and at the end of an epoch modifies the weights.

In addition, the data is presented to the MLP network rotationally. This means that the MLP network will see the data sequentially from time zero. In addition, it will see each pattern one time before updating the weights. This seems to increase the network's chances of detecting

continuous patterns in the data. Since the stock market is so dependent on trends, it was determined that rotational presentation would give the network a fair advantage in prediction. This particular software package will only allow rotational pattern presentation when batch updates are done, mainly because with random presentation, it does not guarantee that every pattern will be chosen an equal number of times during a training epoch.

The first goal of the research is to train and test successfully a backpropagation MLP neural network. Second, a Kohonen network will cluster the input data from the first network to derive confidence table for future predictions.

Before being input into the networks for training or clustering, the input data is preprocessed automatically by NeuroShell 2. The software computes the minimum and maximum values for each feature and normalizes the data between 0 and 1 based on these values.

Test Method

The first step in conducting tests was to successfully train a backpropagation MLP neural network. Successful training was complete when the training set error clearly decreased as training of the net continued. In addition, a test set was used to ensure that training was stopped at the optimum training time; the network weights were saved at

the point when the test set error had reached its minimum value. The production set was extracted before training began and was not used to train or test the network.

The output from this trained network included all of the input features as well as the network's prediction and the actual prediction for each trading day in the training and test sets.

The next step was to cluster the patterns that were used for training and testing in the neural network into 30 categories. After clustering to only 10 categories it appeared that the data could be clustered further and so 30 clusters were selected. Due to the limits of the software more than 30 clusters could not be accomplished.

Once the clustering was complete, each trading day was given a number based on which cluster it was assigned to by the network. The output of the Kohonen network was the list of the input data and features along with a category or cluster number.

Next, the network predictions from the MLP neural network output file were attached to the Kohonen network output file. Now we had a list of each trading day, the MLP network prediction, and a cluster number. The data was then sorted by cluster number; the number of correct and incorrect predictions were totaled for each class (up or

down) in each cluster. A percentage of correct up and down predictions was then figured for each cluster. This comprised the Cluster Confidence Table.

It was hypothesized that given a prediction from the MLP neural network and a cluster number, you can get a more accurate percentage using the confidence table than by using the neural network's accuracy alone. For example, given today's indicators/features, the network predicts the market will go down tomorrow. Today's features when run through the Kohonen network, cluster in Cluster 2. According to the confidence table, when the network predicts down in this cluster, the prediction is 62% accurate based on the number of previously correct predictions in that cluster. If the original network only gave you a 54% accuracy, confidence has been improved in this prediction.

Analysis Method

Two tests were used to determine the success or failure of the this method: one incorporating a production set of 41 randomly picked trading days, and one incorporating a production set of 25 contiguous trading days.

The first test uses of a production set made up of 41 random trading days. These days were run through the trained neural network and then the patterns were clustered using the trained Kohonen network. This yielded a network classification prediction and a cluster number. Using the

confidence table generated from the training and test sets, each trading day was compared to the confidence table. If the percent accuracy for that prediction in that cluster was 60% or greater or 40% or less, a "GO" signal was generated and a hypothetical trade took place. Based on the purchase of real "on the money" stock options, $100 was made for every index point correctly predicted. Similarly, $100 was lost for every index point incorrectly predicted. The use of monetary measures weight the accuracy of our predictions.

In addition, a total dollar comparison was made using only the network predictions (no table consulted). The accuracy percentage will be calculated as well as total dollars won/lost using this method.

Finally, a second test will be run in a similar manner, using a set of 25 contiguous trading days from September and October of 1995. The accuracy achieved both with and without the use of the confidence table will be calculated, as well as the total amount of dollars won/lost in that period. But, in addition, a realistic trading strategy will also be employed to test this method's accuracy.

This Holding Strategy will make the first trade only when the confidence table gives a "GO" signal (shows 60% accuracy or higher) or three days the network gives the same prediction. The option (buy or sell) will be held until either the confidence table gives a "GO" signal in the opposite direction or the neural network predicts two

consecutive days in the opposite direction of the held option. It was our hypothesis that this strategy will result in fewer trading commissions paid out through purchasing and selling options as compared with using the table or using no table at all. In addition, this strategy should yield a higher dollar amount than using the table or using no table at all.

Chapter IV will now illustrate the results of training both the neural and Kohonen networks, as well as the results of the 2 tests outlined above.

IV. Results and Discussion

Random Sample

As noted in Chapter III, the random sample test used a pattern file that included all the trading data from July 1, 1989 through September 28, 1995. The production set of 41 random days was taken from this time period. Originally, these days were randomly selected by the software from this time period. The remainder of the days were then made the training set and the test set was a random 10% taken from the training set. However, no neural network was able to successfully train when the production set was randomly extracted by the software from that time period. Even when the production set was lowered to 30 and then 20 random trading days, the network was unable to train. It was hypothesized that by taking 'chunks' out of this file, it was difficult for the network to find continuous patterns in the input file. Not only was the production set 'chunked' out of the file, but the test set was too.

To test this hypothesis, 41 days were extracted 'evenly' from the pattern file. Specifically, every 38th pattern was extracted to be used as the production set. Every 38th pattern was chosen because it would yield the desired 40 production set patterns. The test set was then

randomly extracted, as before, from the remaining patterns. Once the test set was extracted, the remaining patterns were used as the training set.

Neural Network Training. The network was then trained using batch weight updates. Training was successful, in that the training error continued downward through the 100 epochs of training. Figure 2 shows the training set error of this network over 100 epochs.

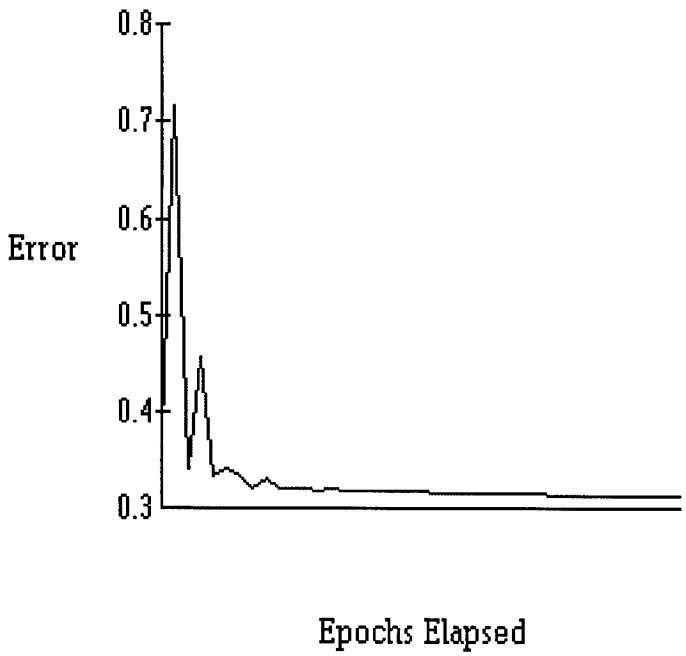

Figure 2. Random Sample Training Set Error Vs. 100 Epochs Elapsed

Clearly successful training occurred as the error continued to decrease over 100 epochs. The test set error for this network is shown in Figure 3. Although the error appears to

be relatively flat, the test set error began to rise very gradually at the 22nd epoch as the network began to "memorize" the training data.

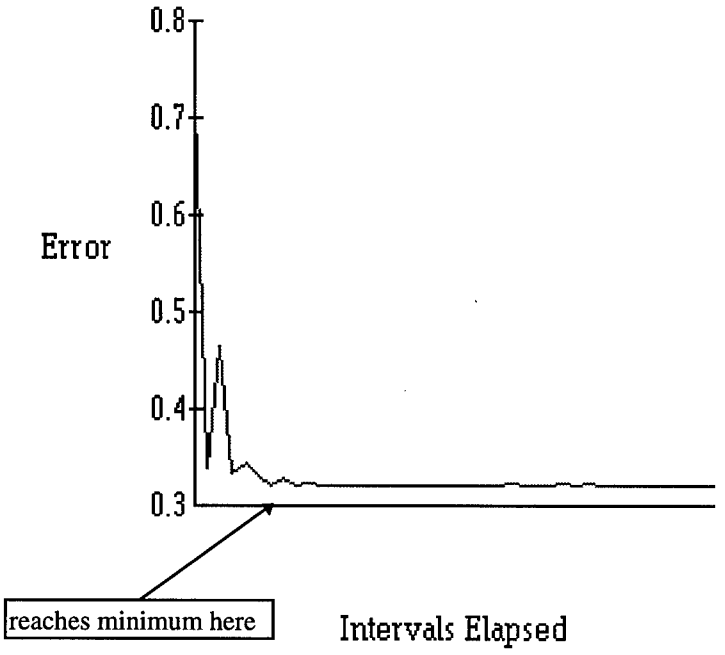

Figure 3. Random Sample Test Set Error Vs. 100 Epochs (Intervals) Elapsed

When this first network was applied to the entire pattern file, the error in prediction was 0.472. Because this rate was relatively high, the correctly predicted trading days were extracted from the file and used to train and test another neural network. It was hypothesized that if there was some correlation in the network's correct versus incorrect predictions, this new network would train perfectly.

Therefore, roughly 52% of the data file was extracted and trained separately. This file was separated into thirds randomly to form the training, test and production sets. The network trained perfectly with overall accuracy against all three sets at 99%. It appeared that the data it was getting right was consistent and learnable or was "lining up" in a way that was helping the network to initially predict this 52% of the data accurately.

A review of the data set from the first neural network revealed that the network was predicting against trend. More specifically, when the net changes were particularly high or low, the network would predict the opposite way (see Table 2). Note that in the first line all three net changes are highly negative indicating a downward trend; the network predicts a 1 or "up" that goes against that trend.

Also of interest in this table is that the network while predicting against the trend, was not always correct (as in line 2 where the OEX actually lost 9.09 points). Other factors besides the 21-day change must be used to gain a correct prediction. In order to see whether it was indeed true that the features were "lining up" in such a way that would almost always yield a correct prediction, a clustering network was employed on the input features to see if they

would cluster in categories where correct predictions were always made.

Table 2

Random Sample Network Predictions (1 = Up)

OEX Delta	1-day net	5-day net	21-day net	Prediction
5.12	-9.09	-18.82	-47.77	1
-9.09	-5.19	-15.52	-39.63	1
9.8	5.12	-15.46	-39.58	1
1.76	-5.79	-12.93	-34.95	1
-5.79	-7.49	-3.21	-33.85	1
-5.19	-6.3	-17.82	-33.54	1
2.39	-0.36	5.47	-32.56	1
-0.36	9.8	0.64	-32.21	1
3.68	-5.79	6.04	-31.61	1
3.93	-4.03	0.95	-29.51	1
-5.79	2.39	16.95	-29.33	1
-0.57	-3.8	-6.46	-28.8	1
-0.68	4.67	1.7	-28.27	1
1.79	-0.68	1.58	-28.19	1
0.34	3.93	4.09	-27.5	1
0.01	-0.57	-5.8	-27.04	1

Kohonen Clustering. The Kohonen clustering network was trained using all data but the original production set of 41 random trading days (every 38th day from the original data set). After 3000 epochs the data had been separated into 30 clusters. A cluster number was assigned to each cluster. Each trading day was then labeled with the cluster number it had been assigned to. Next, the prediction accuracy of MLP neural network was attached to this file. The prediction accuracy was calculated by subtracting the predicted class

from the actual class. A correct prediction was assigned a "0" (1 - 1 = 0 or 0 - 0 = 0). An incorrect up prediction was given a "-1" (0 - 1 = -1) and an incorrect down prediction was a "1" (1 - 0 = 1). This file was trimmed to include only the trading date, the actual class, the cluster number, and the neural network's prediction accuracy (see Table 3).

Development of Confidence Table. The next step was to count the number of correct and incorrect up and down predictions within each cluster. Finally, by dividing the number of correct predictions by the total number of

Table 3

Excerpt From Combined Output File From Random Sample Network

DATE	Up	Cluster	Accuracy
8/23/89	0	8	0
10/17/89	1	3	0
12/12/89	1	18	0
2/6/90	0	29	0
4/2/90	0	11	1
5/25/90	1	18	1
7/20/90	1	13	1
9/13/90	1	13	0
11/6/90	1	27	0
1/2/91	0	13	0
2/26/91	0	17	-1
4/22/91	1	14	0
6/14/91	0	7	0
8/8/91	0	19	1
10/2/91	0	10	-1
11/25/91	1	5	0
1/21/92	1	14	1

predictions within that cluster, a prediction accuracy by cluster and class (up or down) was calculated. This formed the Confidence Table. It was arbitrarily determined that if the confidence in a prediction in a particular cluster was 60% or higher or 40% or lower that a "GO" signal was given, meaning that based on the neural network's prediction and the cluster it was assigned to, a trade should be made based on the prediction. The Confidence Table for the random sample test is shown in Table 4. About 28% of the time, a prediction was made that holds enough confidence for the investor to make a trade.

Results of Using Confidence Table. Next, we used our production set of 41 unseen, unclustered trading days and determined accuracy and money earned. As stated earlier, the money that is earned is based on buying a typical "on the money" stock option in the direction of the prediction. The risk is not calculated since the cost of these options changes daily. It is assumed that if an option is purchased and the index goes in the direction of the option, $100 is earned per index point.

The random production set is first run through the MLP neural network and a predicted classification is given. Next, the random production set is run through the Kohonen network, clustered based on its features, and given a

Table 4

Confidence Table for Random Production Set

	Correct		Incorrect		Confidence		Go/No Go	
CLUSTER	Up	Down	Up	Down	Up	Down	Up	Down
1	30	0	19	0	61%		**Go**	
2	25	0	19	0	57%		N	
3	27	0	18	0	60%		**Go**	
4	31	1	21	1	60%	50%	N	N
5	28	2	23	0	55%	100%	N	**Go**
6	28	0	18	0	61%		**Go**	
7	21	6	21	3	50%	67%	N	**Go**
8	23	13	15	16	61%	45%	**Go**	N
9	25	16	24	11	51%	59%	N	N
10	15	16	20	11	43%	59%	N	N
11	28	3	21	6	57%	33%	N	**Go**
12	20	1	18	1	53%	50%	N	N
13	30	4	33	4	48%	50%	N	N
14	25	2	26	2	49%	50%	N	N
15	30	3	23	0	57%	100%	N	**Go**
16	25	4	20	5	56%	44%	N	N
17	6	12	14	10	30%	55%	**Go**	N
18	5	16	10	22	33%	42%	**Go**	N
19	19	10	13	20	59%	33%	N	**Go**
20	8	26	13	25	38%	51%	**Go**	N
21	5	32	12	17	29%	65%	**Go**	**Go**
22	3	16	1	20	75%	44%	**Go**	N
23	0	16	2	16	0%	50%	**Go**	N
24	2	17	2	12	50%	59%	N	N
25	8	14	3	18	73%	44%	**Go**	N
26	3	20	5	10	38%	67%	**Go**	**Go**
27	4	13	1	14	80%	48%	**Go**	N
28	3	26	2	23	60%	53%	**Go**	N
29	9	12	12	6	43%	67%	N	**Go**
30	20	3	20	3	50%	50%	N	N
					28% of time "GO"			

cluster number. A comparison is made to the Confidence table and it is determined whether a trade is to be made; if it is, the dollar amount earned or lost is listed. In addition, the dollar amount earned or lost without using the

table is also listed. The total dollars earned or lost and the prediction accuracy is listed in Table 5.

By using the Confidence Table, not only was accuracy improved over 20%, but $750 are earned, whereas by depending strictly on the network predictions $185 are lost.

Contiguous Sample

The second test was run using a production set of 25 contiguous days from September and October. Since the test on random days was successful, a test on a group of adjacent days was tested to see if 'real world' use would give similar results. The same procedure was used as in the random sample test. The only addition was to test a "holding strategy" in conjunction with the confidence table. The results will follow.

Neural Network Training. The MLP neural network for the contiguous sample test was trained. The trading days used in the production set were September 18, 1995 through October 21, 1995. This MLP network trained similarly to the MLP network trained for the random sample test. The training set error versus 100 epochs elapsed for this network is shown in Figure 4. The test set error versus 100 epochs elapsed for this network is shown in Figure 5. Clearly the network trained successfully according to our criteria described earlier.

Table 5

Random Sample Test Results

DATE	Actual S&P 100 Change	Network Prediction	Cluster #	GO Signal	$ With Table	$ Without Table
08/23/89	-0.91	1	8	GO	($91)	($91)
10/17/89	5.58	1	3	GO	$558	$558
12/12/89	0.57	1	18			$57
02/06/90	-0.64	1	29			($64)
04/02/90	-2.77	1	11			($277)
05/25/90	0.24	0	18			($24)
07/20/90	0.3	1	13			$30
09/13/90	0.82	1	13			$82
11/06/90	1.74	0	27			($174)
01/02/91	-0.32	1	13			($32)
02/26/91	-1.21	0	17			$121
04/22/91	1.51	1	14			$151
06/14/91	-3.94	1	7			($394)
08/08/91	0	0	19	GO	$0	$0
10/02/91	-3.94	1	10			($394)
11/25/91	2.33	1	5			$233
01/21/92	4.29	0	14			($429)
03/16/92	2.73	1	30			$273
05/08/92	2.67	0	18			($267)
07/02/92	2.04	0	25			($204)
08/26/92	-0.9	1	5			($90)
10/20/92	-0.13	1	21	GO	($13)	($13)
12/14/92	0.11	1	16			$11
02/08/93	-1.84	0	21	GO	$184	$184
04/02/93	0.89	1	14			$89
05/28/93	2.83	0	17			($283)
07/23/93	2.49	1	8	GO	$249	$249
09/16/93	-0.42	0	11	GO	$42	$42
11/09/93	3.19	1	5			$319
01/04/94	0.81	1	17	GO	$81	$81
02/28/94	-2.05	1	6	GO	($205)	($205)
04/22/94	5.26	1	30			$526
06/17/94	-2.49	1	13			($249)
08/11/94	2.7	1	13			$270
10/05/94	-1.84	1	1	GO	($184)	($184)
11/29/94	-1.85	1	30			($185)
01/24/95	1.29	1	3	GO	$129	$129
03/20/95	-1.17	0	17			$117
05/12/95	1.85	0	22			($185)
07/07/95	0.89	0	24			($89)
08/30/95	1.26	1	9			$126
			Total:		**$750**	**($185)**
					73%	**51%**
					Accurate	**Accurate**

The error for the training set and test set combined was 45%. This network was 55% accurate overall. This rate is not much different from the first network. The slightly better rate is most likely due to fewer holes in the data since the production set was not "sliced" out of the original data file as in the random sample test network.

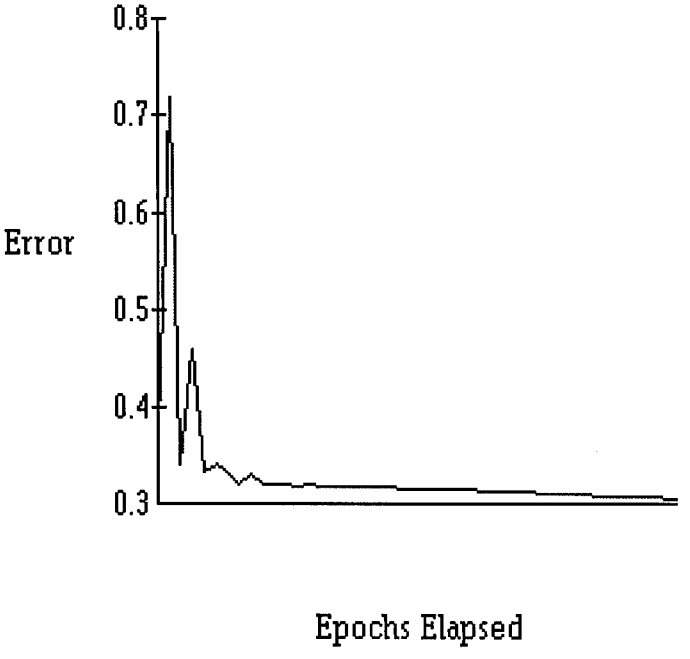

Figure 4. Contiguous Sample Training Set Error Vs. 100 Epochs Elapsed

The networks appear to be sensitive to missing spaces in time in the training set.

The contiguous production set was applied to the MLP network. The result was an error rate of 36%. Therefore, 64% of our trades using only the network will be accurate.

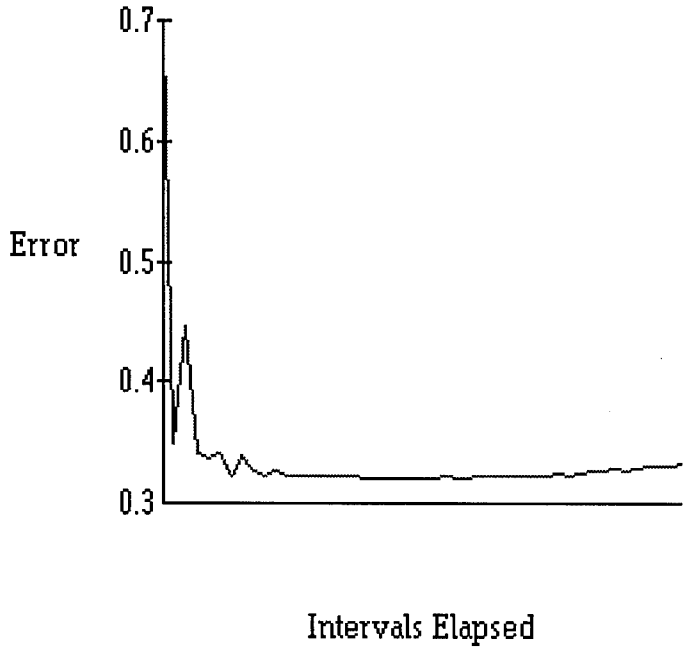

Figure 5. Contiguous Sample Test Set Error Vs. 100 Epochs (Intervals) Elapsed

It should be noted that this is unusually high since the total neural network error is 55%.

Kohonen Clustering. The training and test sets above were combined and used as input into a Kohonen clustering network. The data was clustered into 30 categories after 3000 epochs. As before, the clusters were assigned numbers and each trading day was labeled according to the cluster it was distributed to. The output from the contiguous sample MLP neural network was then attached to this output file. As we did in the random sample test, the correct and incorrect predictions were tallied within each cluster and

an accuracy percentage was given to each class (up or down) within each cluster.

Development of the Confidence Table. The Confidence Table was developed for the contiguous sample test and is located in Table 6.

Table 6

Confidence Table for Contiguous Sample Test

	Correct		Incorrect		Confidence		Go/No Go	
CLUSTER	Up	Down	Up	Down	Up	Down	Up	Down
1	30		19		61%		GO	
2	24		22		52%			
3	29		17		63%		GO	
4	30	1	22	1	58%	50%		
5	28	3	17	0	62%	100%	GO	GO
6	29	3	23	3	56%	50%		
7	20	14	13	7	61%	67%	GO	GO
8	14	16	12	22	54%	42%		
9	25	14	27	8	48%	64%		GO
10	14	15	13	17	52%	47%		
11	19	8	16	1	54%	89%		GO
12	26	12	21	7	55%	63%		GO
13	27	12	16	2	63%	86%	GO	GO
14	29	8	19	1	60%	89%	GO	GO
15	24	3	22	10	52%	23%		GO opp
16	6	19	9	14	40%	58%	GO opp	
17	5	20	7	22	42%	48%		
18	18	18	15	26	55%	41%		
19	9	25	6	30	60%	45%	GO	
20	5	23	8	21	38%	52%	GO opp	
21	8	26	9	12	47%	68%		GO
22	0	15	1	17	0%	47%	GO opp	
23	3	22	2	14	60%	61%	GO	GO
24	3	23	0	16	100%	59%	GO	
25	6	12	3	13	67%	48%	GO	
26	3	16	0	13	100%	55%	GO	
27	4	19	5	19	44%	50%		
28	9	17	11	17	45%	50%		
29	14	6	9	3	61%	67%	GO	GO
30	8	17	13	10	38%	63%	GO opp	GO
						36% of time GO		

Results of Using Confidence Table. The Confidence Table for the contiguous sample test was applied to the production set. The results of using the table and not using the table are shown in Table 7.

Table 7

Results of Contiguous Sample Test

DATE	Actual S&P 100 Change	Cluster #	Network Prediction	GO Signal	$ With Table	$ Without Table
9/18/95	0.93	17	0			($93)
9/19/95	4	16	0			($400)
9/20/95	-3.26	16	0			$326
9/21/95	-0.78	14	0	GO	$78	$78
9/22/95	1.17	15	0	GO UP	$117	($117)
9/25/95	-0.22	3	0			$22
9/26/95	-0.76	3	1	GO	($76)	($76)
9/27/95	3.89	4	1			$389
9/28/95	-1.45	7	0	GO	$145	$145
9/29/95	-2.94	29	1	GO	($294)	($294)
10/2/95	1.04	27	0			($104)
10/3/95	-0.08	9	0	GO	$8	$8
10/4/95	0.8	9	1			$80
10/5/95	0.33	12	1			$33
10/6/95	-3.99	15	0	GO UP	($399)	$399
10/9/95	-0.02	2	0			$2
10/10/95	1.15	2	1			$115
10/11/95	3.34	7	1	GO	$334	$334
10/12/95	1.32	29	1	GO	$132	$132
10/13/95	-0.89	28	0			$89
10/16/95	3.78	27	0			($378)
10/17/95	0.63	26	0			($63)
10/18/95	4.16	27	0			($416)
10/19/95	-3.52	20	0			$352
10/20/95	-2.4	13	0	GO	$240	$240
				TOTAL:	**$285**	**$803**
					70%	**64%**
					Accurate	**Accurate**

It is important to note that although accuracy does improve when the Confidence Table is used, the dollar amount

earned goes down from $803 to $285. Further studies need to be conducted which will pinpoint large move accuracy rather than straight accuracy. For example, one would prefer to accurately predict two large moves and miss ten small moves if it would net more money. So, although the confidence table method will improve confidence in a prediction, it cannot determine whether the index move will be large enough to make an investment worthwhile.

Although roughly $600 more is earned without using the confidence table, transactions fees will be quite high because of daily trading. There are approximately eight changes of position using the network predictions and seven changes of position using the confidence table.

Holding Strategy. In real trading, it is advisable to only enter and exit the market as absolutely needed due to the high costs of transactions. Therefore, rather than continuously buying and selling options according to the network or the network and the table, a "holding strategy" was applied where options were held until either a "GO" signal was generated from the Confidence Table in the opposite direction or two consecutive network predictions were made in the opposite direction. The results of using the "holding strategy" discussed above are shown in Table 8.

Table 8

Results of Holding Strategy

DATE	Action Taken	$
9/18/95	None	$0
9/19/95	None	$0
9/20/95	Buy a "Put" (2 consecutive days down)	$326
9/21/95	Hold "Put"	$78
9/22/95	Sell "Put", Buy "Call"	$117
9/25/95	Hold "Call"	($22)
9/26/95	Hold "Call"	($76)
9/27/95	Hold "Call"	$389
9/28/95	Sell "Call", Buy "Put"	$145
9/29/95	Sell "Put", Buy "Call"	($294)
10/2/95	Hold "Call"	$104
10/3/95	Sell "Call", Buy "Put"	$8
10/4/95	Hold "Put"	($80)
10/5/95	Sell "Put"	$0
10/6/95	Buy "Call"	($399)
10/9/95	Hold "Call"	($2)
10/10/95	Hold "Call"	$115
10/11/95	Hold "Call"	$334
10/12/95	Hold "Call"	$132
10/13/95	Hold "Call"	($89)
10/16/95	Sell "Call"	$0
10/17/95	Buy "Put" (2 consecutive down)	($63)
10/18/95	Hold "Put"	($416)
10/19/95	Hold "Put"	$352
10/20/95	Hold "Put"	$240
	Total:	$899
	Accuracy:	64%

As we can see from Table 8, the holding strategy resulted in seven position changes, but earnings of $899 are higher than either the earnings from using the network or the network with the confidence table. Also, accuracy is the same as that achieved using the neural network alone, but with higher earnings.

Summary

Two tests were outlined to investigate the value of using a neural network with and without a Confidence Table in making predictions of the S&P 100 daily index change; one test used a random sample and the other used a contiguous sample. An MLP neural network was successfully trained for each sample. Next, a Kohonen clustering network was used to cluster the inputs that were used in training the MLP neural networks. The predictive accuracy of the MLP neural network was examined within each cluster and a percentage accuracy or confidence was calculated for each class in each cluster. A confidence table was developed that signaled when a trade should be made based on the percent accuracy within a cluster. Both tests showed that accuracy was improved when the confidence table was used. In addition, using a holding strategy with the contiguous data set, earnings could also be increased.

V. Conclusion and Recommendations

Introduction

The purpose of this research was to identify a method for improving accuracy and confidence in predicting a nonlinear time series when using a neural network. The time series that was used for testing was the S&P 100 stock index. A method that combines the use of a backpropagation MLP neural network and a Kohonen clustering network was examined. Tests of a random sample and a contiguous sample showed that accuracy could be improved by using a confidence table derived from the Kohonen clusters and neural network predictions. This chapter will provide a summary of the results of these tests and draw conclusions based on the results. Potential contributions to the fields of neural network predictions and technical investment analysis will conclude this chapter.

Summary and Discussion of Results

Based on the results of both tests it appears that prediction accuracy can be increased with the use of a confidence table in conjunction with the predictions of a neural network. Before an investment decision is made, consulting the confidence table can improve the accuracy reading of the original neural network prediction, which is

bound by the network's overall prediction accuracy. By clustering the features using a Kohonen clustering network, predictions within clusters can be identified as better or worse depending on which cluster the trading day is assigned to.

Of particular interest, was that in the case contiguous sample, although prediction accuracy increased, dollar earnings did not. Thus, although a network's accuracy implies getting a certain number of trading days "right", it does not guarantee high earnings.

Contributions

This research offers a novel and unique technique for improving neural network prediction accuracy, as well as for adding confidence percentages to predictions. More tests should be conducted to determine the generalizability of this technique to both random and contiguous data sets of nonlinear time series. Finally, by incorporating dollar amounts earned and lost using a neural technique, another crucial dimension is added to the measurement of prediction accuracy.

Appendix A. Data and Software References

Data Source

Decision Point, Incorporated
P.O. Box 7340
Redlands, CA 92375-0340

Fax: (909) 798-7491
Electronic Mail: CarlS16@aol.com

Downloads are available directly by logging onto America Online and going to Keyword "DP".

Software Source

NeuroShell 2

Ward Systems Group, Incorporated
Executive Park West
5 Hillcrest Drive
Frederick, MD 21702

Phone: (301) 662-7950
Fax: (301) 662-5666

Bibliography

Azoff, E.M. Neural Network Time Series Forecasting of Financial Markets. England: John Wiley & Sons, Ltd., 1994.

Barr, Dean S. and Ganesh Mani. "Using Neural Nets to Manage Investments," AI Expert, 9: 16-22 (February 1994).

Decision Point, Incorporated. Decision Point Timing and Charts Forum, America Online, Vienna VA: 1995.

Elgin, Peggie R. "Pioneers Try 'Neural Networks' to Pick Pension Stocks," Corporate Cashflow, 15: 5-6 (July 1994).

Freisleben, Bernd. "Stock Market Prediction with Backpropagation Networks," Proceedings of the Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: 5th International Conference, IEA/AIE-92, Paderborn, Germany, June 9-12, 1992. 451-460. Germany: Springer-Verlag, 1992.

Haykin, Simon. Neural Networks: A Comprehensive Foundation. Canada: Maxwell Macmillan Canada, Inc., 1994.

Longinow, Nicholas E. "Predicting Pilot Look-Angel With a Radial Basis Function Network," IEEE Transactions on Systems, Man and Cybernetics, 24(10): 1511-1518 (1994).

Mockler, Robert J. and D.G. Dologite. An Introduction to Expert Systems: Knowledge-based Systems. New York: Macmillan Publishing Company, 1992.

Muller, B. and Joachim Reinhardt. Neural Networks: An Introduction. Germany: Springer-Verlag Berlin Heidelberg, 1990.

Murray, Dan. "Tuning Neural Networks with Genetic Algorithms," AI Expert, 9: 27-43 (June 1994).

Nelson, M. and W. T. Illingworth. A Practical Guide to Neural Nets. Reading MA: Addison-Wesley Publishing Company, Inc., 1990.

Refenes, Apostolos N., Achileas Zapranis, and Gavin Francis. "Stock Performance Modeling Using Neural Networks: A Comparative Study with Regression Models," Neural Networks: The Official Journal of the International Neural Network Society, 7: 375-388 (1994).

Rogers, Steven K. and Matthew Kabrisky. An Introduction to Biological and Artificial Neural Networks for Pattern Recognition. Bellingham WA: SPIE Optical Engineering Press, 1991.

Ruck, Dennis, et al. "Feature Selection Using a Multilayer Perceptron," Journal of Neural Network Computing, 2(2): 40-48 (1990).

Rumelhart, D.E. "The Basic Ideas in Neural Networks", Communications of the ACM, 37(3): 87 (March 1994).

Ward Systems Group. Neuroshell 2. User's Manual. Frederick MD: Ward Systems Group, Incorporated, November 1993.

Werbos, P. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. PhD dissertation. Harvard University, 1974.

Vita

Erin S. Heim [redacted]. She graduated from Redlands High School in 1984 and entered undergraduate studies at the University of Southern California. She graduated with a Bachelor of Arts degree in Mathematics in May 1988. She married Richard A. Heim of Brooklyn, New York on June 11, 1988. She received a commission in the Air Force on July 25, 1988 upon graduation from the Reserve Officer Training Corps. Her first assignment was at Los Angeles AFB, California as an acquisition project officer for the Strategic Defense Initiative Programs. She gave birth to her first daughter, Alexandria Devon Heim, on September 23, 1992. Her second assignment was at Wright-Patterson AFB, Ohio as Chief, Personnel Requirements, Air Force Materiel Command. In May 1994, she entered the School of Logistics and Acquisition Management, Air Force Institute of Technology and received a Master of Science degree in Information Resource Management on December 19, 1995.

[redacted]